Emulador Cortex-M0

Generado por Doxygen 1.8.10

Domingo, 18 de Octubre de 2015 15:31:05

Índice general

1	Proy	ecto 1:	Emulado	or Cortex-M0	1
2	Índic	ce de es	structura (de datos	3
	2.1	Estruc	tura de da	atos	3
3	Indic	ce de ar	chivos		5
	3.1	Lista d	e archivos	s	5
4	Doci	umenta	ción de la	as estructuras de datos	7
	4.1	Refere	ncia de la	a Estructura ins_t	7
		4.1.1	Docume	entación de los campos	7
			4.1.1.1	array	7
	4.2	Refere	ncia de la	a Estructura instruction_t	7
		4.2.1	Docume	entación de los campos	7
			4.2.1.1	mnemonic	7
			4.2.1.2	op1_type	8
			4.2.1.3	op1_value	8
			4.2.1.4	op2_type	8
			4.2.1.5	op2_value	8
			4.2.1.6	op3_type	8
			4.2.1.7	op3_value	8
			4.2.1.8	registers_list	8
	4.3	Refere	ncia de la	a Estructura port_t	8
		4.3.1	Docume	entación de los campos	8
			4.3.1.1	DDR	8
			4.3.1.2	Interrupts	8
			4.3.1.3	PIN	8
			4.3.1.4	Pins	8
			4.3.1.5	PORT	8
5	Doci	umenta	ción de a	ırchivos	9
	5.1	Refere	ncia del A	Archivo decoder.c	9
		E 1 1	Dooumo	antoción do los Idefinos!	C

IV ÍNDICE GENERAL

		5.1.1.1	PC	9
	5.1.2	Documer	ntación de las funciones	9
		5.1.2.1	countLines(FILE *fp)	9
		5.1.2.2	decodeInstruction(instruction_t instruction, uint32_t *dir_reg, char *dir_flags, uint8_t *SRAM, uint16_t *dec)	9
		5.1.2.3	getInstruction(char *instStr)	10
		5.1.2.4	readFile(char *filename, ins_t *instructions)	11
	5.1.3	Documer	ntación de las variables	11
		5.1.3.1	data	11
		5.1.3.2	dec	11
5.2	Refere	ncia del Ar	rchivo decoder.h	11
	5.2.1	Documer	ntación de las funciones	11
		5.2.1.1	countLines(FILE *fp)	11
		5.2.1.2	decodeInstruction(instruction_t instruction, uint32_t *dir_reg, char *dir_flags, uint8_t *SRAM, uint16_t *dec)	11
		5.2.1.3	getInstruction(char *instStr)	11
		5.2.1.4	readFile(char *filename, ins_t *instructions)	12
5.3	Refere	ncia del Ar	rchivo imprimir.c	12
	5.3.1	Documer	ntación de las funciones	12
		5.3.1.1	valor_registro(uint32_t valores[])	12
5.4	Refere	ncia del Ar	rchivo imprimir.h	12
	5.4.1	Documer	ntación de las funciones	13
		5.4.1.1	valor_registro(uint32_t valores[])	13
5.5	Refere	ncia del Ar	rchivo instruc_desplazamiento.c	13
	5.5.1	Documer	ntación de los 'defines'	14
		5.5.1.1	C	14
		5.5.1.2	$N \ \dots $	14
		5.5.1.3	v	14
		5.5.1.4	z	14
	5.5.2	Documer	ntación de las funciones	15
		5.5.2.1	ASR(uint32_t Rdn, uint32_t Rm, char *dir_flags)	15
		5.5.2.2	BIC(uint32_t Rdn, uint32_t Rm, char *dir_flags)	15
		5.5.2.3	LSL(uint32_t Rdn, uint32_t Rm, char *dir_flags)	15
		5.5.2.4	LSR(uint32_t Rdn, uint32_t Rm, char *dir_flags)	16
		5.5.2.5	MVN(uint32_t Rdn, char *dir_flags)	16
		5.5.2.6	REV(uint32_t Rdn)	17
		5.5.2.7	REVG(uint32_t Rdn)	17
		5.5.2.8	REVSH(uint32_t Rdn)	17
		5.5.2.9	ROR(uint32_t Rdn, uint32_t Rm, char *dir_flags)	18
		5.5.2.10	RSB(uint32_t Rdn, char *dir_flags)	18

ÍNDICE GENERAL v

5.6	Refere	ncia del Ar	rchivo instruc_desplazamiento.h	18
	5.6.1	Documer	ntación de las funciones	20
		5.6.1.1	ASR(uint32_t Rdn, uint32_t Rm, char *dir_flags)	20
		5.6.1.2	BIC(uint32_t Rdn, uint32_t Rm, char *dir_flags)	20
		5.6.1.3	LSL(uint32_t Rdn, uint32_t Rm, char *dir_flags)	20
		5.6.1.4	LSR(uint32_t Rdn, uint32_t Rm, char *dir_flags)	22
		5.6.1.5	MVN(uint32_t Rdn, char *dir_flags)	22
		5.6.1.6	REV(uint32_t Rdn)	23
		5.6.1.7	REVG(uint32_t Rdn)	23
		5.6.1.8	REVSH(uint32_t Rdn)	23
		5.6.1.9	ROR(uint32_t Rdn, uint32_t Rm, char *dir_flags)	24
		5.6.1.10	RSB(uint32_t Rdn, char *dir_flags)	24
5.7	Refere	ncia del Ar	rchivo instrucciones.c	24
	5.7.1	Documer	ntación de los 'defines'	27
		5.7.1.1	C	27
		5.7.1.2	LR	27
		5.7.1.3	$N \ \dots $	27
		5.7.1.4	PC	28
		5.7.1.5	SP	28
		5.7.1.6	V	28
		5.7.1.7	Z	28
	5.7.2	Documer	ntación de las funciones	28
		5.7.2.1	ADC(uint32_t Rn, uint32_t Rm, char *dir_flags)	28
		5.7.2.2	ADD(uint32_t Rn, uint32_t Rm, char *dir_flags)	28
		5.7.2.3	AND(uint32_t Rn, uint32_t Rm, char *dir_flags)	29
		5.7.2.4	$bitcount(uint8_t*R) $	29
		5.7.2.5	CMN(uint32_t Rn, uint32_t Rm, char *dir_flags)	29
		5.7.2.6	CMP(uint32_t Rn, uint32_t Rm, char *dir_flags)	30
		5.7.2.7	EOR(uint32_t Rn, uint32_t Rm, char *dir_flags)	30
		5.7.2.8	flag_C(uint32_t Rn, uint32_t Rm, uint32_t Rd, char *dir_flag_C)	30
		5.7.2.9	flag_N(uint32_t Rd, char *dir_flag_N)	31
		5.7.2.10	flag_V(uint32_t Rn, uint32_t Rm, uint32_t Rd, char *dir_flag_V)	31
		5.7.2.11	flag_Z(uint32_t Rd, char *dir_flag_Z)	31
		5.7.2.12	flags(uint32_t Rn, uint32_t Rm, uint32_t Rd, char *dir_flags)	32
		5.7.2.13	LDR(uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	32
		5.7.2.14	LDRB(uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	32
		5.7.2.15	LDRH(uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	33
		5.7.2.16	LDRSB(uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	34
		5.7.2.17	LDRSH(uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	34
		5.7.2.18	MOV(uint32_t Rn, char *dir_flags)	34

VI ÍNDICE GENERAL

		5.7.2.19	MUL(uint32_t Rn, uint32_t Rm, char *dir_flags)	35
		5.7.2.20	NOP(uint32_t *dir_reg)	35
		5.7.2.21	NVIC(uint8_t *IRQ, uint8_t *SRAM, uint32_t *dir_reg, char *dir_flags)	35
		5.7.2.22	ORR(uint32_t Rn, uint32_t Rm, char *dir_flags)	36
		5.7.2.23	POP(uint8_t *SRAM, uint32_t *dir_reg, uint8_t *R_activos)	36
		5.7.2.24	POPI(uint8_t *SRAM, uint32_t *dir_reg, char *dir_flags)	36
		5.7.2.25	PUSH(uint8_t *SRAM, uint32_t *dir_reg, uint8_t *R_activos)	36
		5.7.2.26	PUSHI(uint8_t *SRAM, uint32_t *dir_reg, char *dir_flags)	37
		5.7.2.27	SBC(uint32_t Rn, uint32_t Rm, char *dir_flags)	37
		5.7.2.28	STR(uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	37
		5.7.2.29	STRB(uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	37
		5.7.2.30	STRH(uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	38
		5.7.2.31	SUB(uint32_t Rn, uint32_t Rm, char *dir_flags)	38
		5.7.2.32	TST(uint32_t Rn, uint32_t Rm, char *dir_flags)	38
5.8	Refere	ncia del Ar	chivo instrucciones.h	39
	5.8.1	Documer	ntación de las funciones	41
		5.8.1.1	ADC(uint32_t Rn, uint32_t Rm, char *dir_flags)	41
		5.8.1.2	ADD(uint32_t Rn, uint32_t Rm, char *dir_flags)	42
		5.8.1.3	AND(uint32_t Rn, uint32_t Rm, char *dir_flags)	42
		5.8.1.4	$bitcount(uint8_t*R) \qquad \dots \qquad \dots \qquad \dots \qquad \dots$	43
		5.8.1.5	CMN(uint32_t Rn, uint32_t Rm, char *dir_flags)	43
		5.8.1.6	CMP(uint32_t Rn, uint32_t Rm, char *dir_flags)	43
		5.8.1.7	EOR(uint32_t Rn, uint32_t Rm, char *dir_flags)	43
		5.8.1.8	flag_C(uint32_t Rn, uint32_t Rm, uint32_t Rd, char *dir_flag_C)	45
		5.8.1.9	flag_N(uint32_t Rd, char *dir_flag_N)	45
		5.8.1.10	flag_V(uint32_t Rn, uint32_t Rm, uint32_t Rd, char *dir_flag_V)	45
		5.8.1.11	flag_Z(uint32_t Rd, char *dir_flag_Z)	46
		5.8.1.12	flags(uint32_t Rn, uint32_t Rm, uint32_t Rd, char *dir_flags)	46
		5.8.1.13	LDR(uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	46
		5.8.1.14	LDRB(uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	47
		5.8.1.15	LDRH(uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	47
		5.8.1.16	LDRSB(uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	47
		5.8.1.17	LDRSH(uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	47
		5.8.1.18	MOV(uint32_t Rn, char *dir_flags)	48
		5.8.1.19	MUL(uint32_t Rn, uint32_t Rm, char *dir_flags)	48
		5.8.1.20	NOP(uint32_t *dir_reg)	49
		5.8.1.21	NVIC(uint8_t *IRQ, uint8_t *SRAM, uint32_t *dir_reg, char *dir_flags)	49
		5.8.1.22	ORR(uint32_t Rn, uint32_t Rm, char *dir_flags)	49
		5.8.1.23	POP(uint8_t *SRAM, uint32_t *dir_reg, uint8_t *R_activos)	49
		5.8.1.24	POPI(uint8_t *SRAM, uint32_t *dir_reg, char *dir_flags)	50

ÍNDICE GENERAL VII

			5.8.1.25	PUSH(uint8_t *SRAM, uint32_t *dir_reg, uint8_t *R_activos)	50
			5.8.1.26	PUSHI(uint8_t *SRAM, uint32_t *dir_reg, char *dir_flags)	50
			5.8.1.27	SBC(uint32_t Rn, uint32_t Rm, char *dir_flags)	50
			5.8.1.28	STR(uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	51
			5.8.1.29	STRB(uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	51
			5.8.1.30	STRH(uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t *SRAM)	51
			5.8.1.31	SUB(uint32_t Rn, uint32_t Rm, char *dir_flags)	51
			5.8.1.32	TST(uint32_t Rn, uint32_t Rm, char *dir_flags)	52
5	5.9	Referer	ncia del Ar	chivo io.c	52
		5.9.1	Documer	ntación de las funciones	53
			5.9.1.1	changePinPortA(uint8_t pin, uint8_t value)	53
			5.9.1.2	changePinPortB(uint8_t pin, uint8_t value)	53
			5.9.1.3	initlO(void)	53
			5.9.1.4	IOAccess(uint8_t address, uint8_t *data, uint8_t r_w)	53
			5.9.1.5	$showFrame(int \ x, \ int \ y, \ int \ w, \ int \ h) \ \ . \ \ \ \ \ . \ \ \ \ \ . \$	53
			5.9.1.6	showPorts(void)	53
		5.9.2	Documer	ntación de las variables	54
			5.9.2.1	PORTA	54
			5.9.2.2	PORTB	54
5	5.10	Referer	ncia del Ar	chivo io.h	54
		5.10.1	Documer	ntación de los 'defines'	55
			5.10.1.1	BLUEBLACK	55
			5.10.1.2	HIGH	55
			5.10.1.3	LOW	55
			5.10.1.4	Read	55
			5.10.1.5	REDBLACK	55
			5.10.1.6	WHITEBLACK	55
			5.10.1.7	Write	55
			5.10.1.8	XINIT	55
			5.10.1.9	YINIT	55
		5.10.2	Documer	ntación de las funciones	55
			5.10.2.1	changePinPortA(uint8_t pin, uint8_t value)	55
			5.10.2.2	changePinPortB(uint8_t pin, uint8_t value)	55
			5.10.2.3	initlO(void)	55
			5.10.2.4	IOAccess(uint8_t address, uint8_t *data, uint8_t r_w)	55
			5.10.2.5	$showFrame(int \ x, \ int \ y, \ int \ w, \ int \ h) \ \ . \ \ \ \ . \ \ \ \ \ \ \ \ . \$	56
			5.10.2.6	showPorts(void)	56
		5.10.3	Documer	ntación de las variables	56
			5.10.3.1	IRQ	56
5	5.11	Referer	ncia del Ar	chivo main.c	56

VIII ÍNDICE GENERAL

5.11.1	Documentación de los 'defines'	57
	5.11.1.1 C	57
	5.11.1.2 LR	57
	5.11.1.3 N	57
	5.11.1.4 PC	57
	5.11.1.5 SP	57
	5.11.1.6 TAM_SRAM	57
	5.11.1.7 V	57
	5.11.1.8 Z	57
5.11.2	Documentación de las funciones	57
	5.11.2.1 main(void)	57
5.12 Refere	ncia del Archivo README.md	57
5.13 Refere	ncia del Archivo salto.c	57
5.13.1	Documentación de los 'defines'	60
	5.13.1.1 C	60
	5.13.1.2 LR	60
	5.13.1.3 N	60
	5.13.1.4 PC	60
	5.13.1.5 V	60
	5.13.1.6 Z	60
5.13.2	Documentación de las funciones	60
	5.13.2.1 B(uint32_t label, uint32_t *dir_reg)	60
	5.13.2.2 BAL(uint32_t label, uint32_t *dir_reg)	61
	5.13.2.3 BCC(uint32_t label, uint32_t *dir_reg, char *dir_flags)	61
	5.13.2.4 BCS(uint32_t label, uint32_t *dir_reg, char *dir_flags)	61
	5.13.2.5 BEQ(uint32_t label, uint32_t *dir_reg, char *dir_flags)	62
	5.13.2.6 BGE(uint32_t label, uint32_t *dir_reg, char *dir_flags)	62
	5.13.2.7 BGT(uint32_t label, uint32_t *dir_reg, char *dir_flags)	62
	5.13.2.8 BHI(uint32_t label, uint32_t *dir_reg, char *dir_flags)	63
	5.13.2.9 BL(uint32_t label, uint32_t *dir_reg)	63
	5.13.2.10 BLE(uint32_t label, uint32_t *dir_reg, char *dir_flags)	63
	5.13.2.11 BLS(uint32_t label, uint32_t *dir_reg, char *dir_flags)	64
	5.13.2.12 BLT(uint32_t label, uint32_t *dir_reg, char *dir_flags)	64
	5.13.2.13 BMI(uint32_t label, uint32_t *dir_reg, char *dir_flags)	64
	5.13.2.14 BNE(uint32_t label, uint32_t *dir_reg, char *dir_flags)	65
	5.13.2.15 BPL(uint32_t label, uint32_t *dir_reg, char *dir_flags)	65
	5.13.2.16 BVC(uint32_t label, uint32_t *dir_reg, char *dir_flags)	65
	5.13.2.17 BVS(uint32_t label, uint32_t *dir_reg, char *dir_flags)	66
	5.13.2.18 BX(uint32_t *dir_reg)	66
5.14 Refere	ncia del Archivo salto.h	66

ÍNDICE GENERAL IX

	5.14.1	Document	ación de las funciones	68
		5.14.1.1	B(uint32_t label, uint32_t *dir_reg)	68
		5.14.1.2	BAL(uint32_t label, uint32_t *dir_reg)	70
		5.14.1.3	BCC(uint32_t label, uint32_t *dir_reg, char *dir_flags)	70
		5.14.1.4	BCS(uint32_t label, uint32_t *dir_reg, char *dir_flags)	70
		5.14.1.5	BEQ(uint32_t label, uint32_t *dir_reg, char *dir_flags)	71
		5.14.1.6	BGE(uint32_t label, uint32_t *dir_reg, char *dir_flags)	71
		5.14.1.7	BGT(uint32_t label, uint32_t *dir_reg, char *dir_flags)	71
		5.14.1.8	BHI(uint32_t label, uint32_t *dir_reg, char *dir_flags)	72
		5.14.1.9	BL(uint32_t label, uint32_t *dir_reg)	72
		5.14.1.10	BLE(uint32_t label, uint32_t *dir_reg, char *dir_flags)	72
		5.14.1.11	BLS(uint32_t label, uint32_t *dir_reg, char *dir_flags)	73
		5.14.1.12	BLT(uint32_t label, uint32_t *dir_reg, char *dir_flags)	73
		5.14.1.13	BMI(uint32_t label, uint32_t *dir_reg, char *dir_flags)	73
		5.14.1.14	BNE(uint32_t label, uint32_t *dir_reg, char *dir_flags)	74
		5.14.1.15	BPL(uint32_t label, uint32_t *dir_reg, char *dir_flags)	74
		5.14.1.16	BVC(uint32_t label, uint32_t *dir_reg, char *dir_flags)	74
		5.14.1.17	BVS(uint32_t label, uint32_t *dir_reg, char *dir_flags)	75
		5.14.1.18	BX(uint32_t *dir_reg)	75
5.15	Referer	ncia del Arc	chivo test.c	75
	5.15.1	Document	ación de los 'defines'	76
		5.15.1.1	C	76
		5.15.1.2	N	76
		5.15.1.3	V	76
		5.15.1.4	Z	76
	5.15.2	Document	ación de las funciones	76
		5.15.2.1	main()	76

Proyecto 1: Emulador Cortex-M0

Emulador Cortex-M0

Materia: Microprocesadores

Codigo destinado a emular el funcionamiento de este procesador.

Autores

John Barahona Heyler Montoya Javier Sierra

Fecha

2015 - II

Repositorio disponible en GitHub: Click Aqui

```
John Alejandro Barahona Pineda - (alejo7053)

Heyler Stivens Montoya Orjuela - (heylermontoya)

Javier Andres Sierra Pineda - (sierral12)
```

Índice de estructura de datos

2.	1	Fetru	ctura	de	datne
6 :	1.	LSHU	Gluia	uc	ualus

Lista de	estructuras	con	una	breve	descri	nción
Lista ac	Collacturas	COLL	una	DICVC	acscri	

<u> </u>	7
$instruction_t \ \dots $	
port_t	8

ndice				

Indice de archivos

3.1. Lista de archivos

Lista de todos los archivos con descripciones breves:

decoder.c	
decoder.h	
imprimir.c	
imprimir.h	12
instruc_desplazamiento.c	
instruc_desplazamiento.h	
instrucciones.c	
instrucciones.h	
io.c	
io.h	54
main.c	
salto.c	
salto.h	
test c	75

6 Indice de archivos

Documentación de las estructuras de datos

4.1. Referencia de la Estructura ins_t

```
#include <decoder.h>
```

Campos de datos

■ char ** array

4.1.1. Documentación de los campos

```
4.1.1.1. char** array
```

La documentación para esta estructura fue generada a partir del siguiente fichero:

decoder.h

4.2. Referencia de la Estructura instruction_t

```
#include <decoder.h>
```

Campos de datos

- char mnemonic [10]
- char op1_type
- char op2_type
- char op3_type
- uint32_t op1_value
- uint32_t op2_value
- uint32_t op3_value
- uint8_t registers_list [16]

4.2.1. Documentación de los campos

4.2.1.1. char mnemonic[10]

```
4.2.1.2. char op1_type
4.2.1.3. uint32_t op1_value
4.2.1.4. char op2_type
4.2.1.5. uint32_t op2_value
4.2.1.6. char op3_type
4.2.1.7. uint32_t op3_value
```

4.2.1.8. uint8_t registers_list[16]

La documentación para esta estructura fue generada a partir del siguiente fichero:

decoder.h

4.3. Referencia de la Estructura port_t

```
#include <io.h>
```

Campos de datos

- uint8_t DDR
- uint8 t PORT
- uint8_t PIN
- uint8_t Pins
- uint8_t Interrupts

4.3.1. Documentación de los campos

- 4.3.1.1. uint8_t DDR
- 4.3.1.2. uint8_t Interrupts
- 4.3.1.3. uint8_t PIN
- 4.3.1.4. uint8_t Pins
- 4.3.1.5. uint8_t PORT

La documentación para esta estructura fue generada a partir del siguiente fichero:

io.h

Documentación de archivos

5.1. Referencia del Archivo decoder.c

```
#include "decoder.h"
```

'defines'

■ #define PC 15

Macro que define la posicion del registro PC en el puntero dir_reg.

Funciones

- void decodeInstruction (instruction_t instruction, uint32_t *dir_reg, char *dir_flags, uint8_t *SRAM, uint16_t
 *dec)
- instruction_t getInstruction (char *instStr)

Obtiene la instrucción separada por partes.

- int readFile (char *filename, ins_t *instructions)
- int countLines (FILE *fp)

Variables

- uint8 t data
- uint16_t * dec

5.1.1. Documentación de los 'defines'

5.1.1.1. #define PC 15

Macro que define la posicion del registro PC en el puntero dir_reg.

5.1.2. Documentación de las funciones

```
5.1.2.1. int countLines ( FILE * fp )
```

5.1.2.2. void decodeInstruction (instruction_t instruction, uint32_t * dir_reg, char * dir_flags, uint8_t * SRAM, uint16_t * dec)

5.1.2.3. instruction_t getInstruction (char * instStr)

Obtiene la instrucción separada por partes.

Parámetros

instStr	cadena que contiene la instrucción.
---------	-------------------------------------

Devuelve

instruction_t la instrucción separada por partes.

- 5.1.2.4. int readFile (char * filename, ins_t * instructions)
- 5.1.3. Documentación de las variables
- 5.1.3.1. uint8_t data
- 5.1.3.2. uint16_t* dec

5.2. Referencia del Archivo decoder.h

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <stdint.h>
#include "instrucciones.h"
#include "instruc_desplazamiento.h"
#include "io.h"
#include "salto.h"
#include <curses.h>
```

Estructuras de datos

- struct ins_t
- struct instruction_t

Funciones

- void decodeInstruction (instruction_t instruction, uint32_t *dir_reg, char *dir_flags, uint8_t *SRAM, uint16_t
 *dec)
- instruction_t getInstruction (char *instStr)

Obtiene la instrucción separada por partes.

- int readFile (char *filename, ins_t *instructions)
- int countLines (FILE *fp)

5.2.1. Documentación de las funciones

```
5.2.1.1. int countLines (FILE * fp )
```

- 5.2.1.2. void decodeInstruction (instruction_t instruction, uint32_t * dir_reg, char * dir_flags, uint8_t * SRAM, uint16_t * dec)
- 5.2.1.3. instruction_t getInstruction (char * instStr)

Obtiene la instrucción separada por partes.

Parámetros

instStr	cadena que contiene la instrucción.	

Devuelve

instruction_t la instrucción separada por partes.

```
5.2.1.4. int readFile ( char * filename, ins_t * instructions )
```

5.3. Referencia del Archivo imprimir.c

```
#include "imprimir.h"
```

Funciones

void valor_registro (uint32_t valores[])

valor_registro

prototipo de la funcion valor_registro que muestra el valor que tienen los registros

5.3.1. Documentación de las funciones

5.3.1.1. void valor_registro (uint32_t valores[])

valor_registro

prototipo de la funcion valor_registro que muestra el valor que tienen los registros

5.4. Referencia del Archivo imprimir.h

```
#include <stdio.h>
#include <stdint.h>
#include <stdlib.h>
#include <curses.h>
```

Funciones

void valor_registro (uint32_t valores[])

valor_registro

prototipo de la funcion valor_registro que muestra el valor que tienen los registros

5.4.1. Documentación de las funciones

5.4.1.1. void valor_registro (uint32_t valores[])

valor_registro

prototipo de la funcion valor_registro que muestra el valor que tienen los registros

5.5. Referencia del Archivo instruc_desplazamiento.c

#include "instruc_desplazamiento.h"

'defines'

■ #define N 0

Macro que define la posicion de la bandera N en el puntero dir_flags.

#define Z 1

Macro que define la posicion de la bandera Z en el puntero dir_flags.

■ #define C 2

Macro que define la posicion de la bandera C en el puntero dir_flags.

■ #define V 3

Macro que define la posicion de la bandera V en el puntero dir_flags.

Funciones

uint32_t LSL (uint32_t Rdn, uint32_t Rm, char *dir_flags)

LSL

prototipo de la funcion de desplazamiento LSL que va a operar los registros.

uint32_t LSR (uint32_t Rdn, uint32_t Rm, char *dir_flags)

LSR

prototipo de la funcion de desplazamiento LSR que va a operar los registros

uint32_t ROR (uint32_t Rdn, uint32_t Rm, char *dir_flags)

ROR

prototipo de la funcion de desplazamiento ROR que va a operar los registros

uint32_t ASR (uint32_t Rdn, uint32_t Rm, char *dir_flags)

ASR

prototipo de la funcion de desplazamiento ASR que va a operar los registros

uint32_t BIC (uint32_t Rdn, uint32_t Rm, char *dir_flags)

BIC

prototipo de la funcion de desplazamiento BIC que va a operar los registros

■ uint32 t MVN (uint32 t Rdn, char *dir flags)

MVN

prototipo de la funcion de desplazamiento MVN que va a operar los registros

uint32_t RSB (uint32_t Rdn, char *dir_flags)

RSB

prototipo de la funcion de desplazamiento RSB que va a operar los registros

uint32_t REV (uint32_t Rdn)

REV

prototipo de la funcion de desplazamiento REV que va a operar los registros

uint32_t REVG (uint32_t Rdn)

REVG

prototipo de la funcion de desplazamiento REV que va a operar los registros

uint32_t REVSH (uint32_t Rdn)

REVSH

prototipo de la funcion de desplazamiento REV que va a operar los registros

5.5.1. Documentación de los 'defines'

5.5.1.1. #define C 2

Macro que define la posicion de la bandera *C en* el puntero dir_flags.

5.5.1.2. #define N 0

Macro que define la posicion de la bandera *N en* el puntero dir_flags.

5.5.1.3. #define V 3

Macro que define la posicion de la bandera *V en* el puntero dir_flags.

5.5.1.4. #define Z 1

Macro que define la posicion de la bandera *Z en* el puntero dir_flags.

5.5.2. Documentación de las funciones

5.5.2.1. uint32_t ASR (uint32_t Rdn, uint32_t Rm, char * dir_flags)

ASR

prototipo de la funcion de desplazamiento ASR que va a operar los registros

Parámetros

Rdn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion ASR

5.5.2.2. uint32_t BIC (uint32_t Rdn, uint32_t Rm, char * dir_flags)

BIC

prototipo de la funcion de desplazamiento BIC que va a operar los registros

Parámetros

Rdn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion BIC

5.5.2.3. uint32_t LSL (uint32_t Rdn, uint32_t Rm, char * dir_flags)

LSL

prototipo de la funcion de desplazamiento LSL que va a operar los registros.

Parámetros

Rdi	operando ingresado
Rn	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion LSL

5.5.2.4. uint32_t LSR (uint32_t Rdn, uint32_t Rm, char * dir_flags)

LSR

prototipo de la funcion de desplazamiento LSR que va a operar los registros

Parámetros

Ro	operando ingresado
R	operando ingresado
dir_flag	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion LSR

5.5.2.5. uint32_t MVN (uint32_t Rdn, char * dir_flags)

MVN

prototipo de la funcion de desplazamiento MVN que va a operar los registros

Parámetros

Rdn	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion MVN

5.5.2.6. uint32_t REV (uint32_t Rdn)

REV

prototipo de la funcion de desplazamiento REV que va a operar los registros

Parámetros

Rdn	operando ingresado

Devuelve

retorna el resultado de la operacion REV

5.5.2.7. uint32_t REVG (uint32_t *Rdn*)

REVG

prototipo de la funcion de desplazamiento REV que va a operar los registros

Parámetros

Rdn	operando ingresado

Devuelve

retorna el resultado de la operacion REVG

5.5.2.8. uint32_t REVSH (uint32_t Rdn)

REVSH

prototipo de la funcion de desplazamiento REV que va a operar los registros

Parámetros

Rdn	operando ingresado

Devuelve

retorna el resultado de la operacion REVSH

5.5.2.9. uint32_t ROR (uint32_t Rdn, uint32_t Rm, char * dir_flags)

ROR

prototipo de la funcion de desplazamiento ROR que va a operar los registros

Parámetros

Rdn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion ROR

5.5.2.10. uint32_t RSB (uint32_t Rdn, char * dir_flags)

RSB

prototipo de la funcion de desplazamiento RSB que va a operar los registros

Parámetros

Rdn	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion RSB

5.6. Referencia del Archivo instruc_desplazamiento.h

```
#include <stdio.h>
#include <stdint.h>
#include <stdlib.h>
#include "instrucciones.h"
```

Funciones

uint32_t LSL (uint32_t Rdn, uint32_t Rm, char *dir_flags)

LSL

prototipo de la funcion de desplazamiento LSL que va a operar los registros.

uint32_t LSR (uint32_t Rdn, uint32_t Rm, char *dir_flags)

LSF

prototipo de la funcion de desplazamiento LSR que va a operar los registros

uint32_t ROR (uint32_t Rdn, uint32_t Rm, char *dir_flags)

ROR

prototipo de la funcion de desplazamiento ROR que va a operar los registros

uint32_t ASR (uint32_t Rdn, uint32_t Rm, char *dir_flags)

ASR

prototipo de la funcion de desplazamiento ASR que va a operar los registros

uint32_t BIC (uint32_t Rdn, uint32_t Rm, char *dir_flags)

BIC

prototipo de la funcion de desplazamiento BIC que va a operar los registros

uint32_t MVN (uint32_t Rdn, char *dir_flags)

MVN

prototipo de la funcion de desplazamiento MVN que va a operar los registros

uint32_t RSB (uint32_t Rdn, char *dir_flags)

RSB

prototipo de la funcion de desplazamiento RSB que va a operar los registros

uint32_t REV (uint32_t Rdn)

REV

prototipo de la funcion de desplazamiento REV que va a operar los registros

uint32_t REVG (uint32_t Rdn)

REVG

prototipo de la funcion de desplazamiento REV que va a operar los registros

uint32_t REVSH (uint32_t Rdn)

REVSH

prototipo de la funcion de desplazamiento REV que va a operar los registros

5.6.1. Documentación de las funciones

5.6.1.1. uint32_t ASR (uint32_t Rdn, uint32_t Rm, char * dir_flags)

ASR

prototipo de la funcion de desplazamiento ASR que va a operar los registros

Parámetros

Rdn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion ASR

5.6.1.2. uint32_t BIC (uint32_t Rdn, uint32_t Rm, char * dir_flags)

BIC

prototipo de la funcion de desplazamiento BIC que va a operar los registros

Parámetros

Rdn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion BIC

5.6.1.3. uint32_t LSL (uint32_t Rdn, uint32_t Rm, char * dir_flags)

LSL

prototipo de la funcion de desplazamiento LSL que va a operar los registros.

Parámetros

Rdn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion LSL

5.6.1.4. uint32_t LSR (uint32_t Rdn, uint32_t Rm, char * dir_flags)

LSR

prototipo de la funcion de desplazamiento LSR que va a operar los registros

Parámetros

Ro	operando ingresado
R	operando ingresado
dir_flag	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion LSR

5.6.1.5. uint32_t MVN (uint32_t Rdn, char * dir_flags)

MVN

prototipo de la funcion de desplazamiento MVN que va a operar los registros

Parámetros

Rdn	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion MVN

5.6.1.6. uint32_t REV (uint32_t Rdn)

REV

prototipo de la funcion de desplazamiento REV que va a operar los registros

Parámetros

Rdn	operando ingresado

Devuelve

retorna el resultado de la operacion REV

5.6.1.7. uint32_t REVG (uint32_t Rdn)

REVG

prototipo de la funcion de desplazamiento REV que va a operar los registros

Parámetros

Rdn	operando ingresado

Devuelve

retorna el resultado de la operacion REVG

5.6.1.8. uint32_t REVSH (uint32_t Rdn)

REVSH

prototipo de la funcion de desplazamiento REV que va a operar los registros

Parámetros

Rdn	operando ingresado

Devuelve

retorna el resultado de la operacion REVSH

5.6.1.9. uint32_t ROR (uint32_t Rdn, uint32_t Rm, char * dir_flags)

ROR

prototipo de la funcion de desplazamiento ROR que va a operar los registros

Parámetros

Rdn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion ROR

5.6.1.10. uint32_t RSB (uint32_t Rdn, char * dir_flags)

RSB

prototipo de la funcion de desplazamiento RSB que va a operar los registros

Parámetros

Rdn	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion RSB

5.7. Referencia del Archivo instrucciones.c

#include "instrucciones.h"

'defines'

■ #define N 0

Macro que define la posicion de la bandera N en el puntero dir_flags.

■ #define Z 1

Macro que define la posicion de la bandera **Z** en el puntero **dir_flags**.

■ #define C 2

Macro que define la posicion de la bandera C en el puntero dir_flags.

#define V 3

Macro que define la posicion de la bandera V en el puntero dir_flags.

■ #define PC 15

Macro que define la posicion del registro PC en el puntero dir_reg.

#define LR 14

Macro que define la posicion del registro LR en el puntero dir_reg.

#define SP 13

Macro que define la posicion del registro SP en el puntero dir_reg.

Funciones

void flags (uint32_t Rn, uint32_t Rm, uint32_t Rd, char *dir_flags)

flags

prototipo de la funcion flags que va a operar los registros de las banderas

void flag_N (uint32_t Rd, char *dir_flag_N)

flag_N

prototipo de la funcion flag_N que va a operar el registro de la bandera N

void flag_Z (uint32_t Rd, char *dir_flag_Z)

flag_Z

prototipo de la funcion flag_Z que va a operar el registro de la bandera Z

■ void flag_C (uint32_t Rn, uint32_t Rm, uint32_t Rd, char *dir_flag_C)

flag_C

prototipo de la funcion flag_C que va a operar el registro de la bandera C

void flag_V (uint32_t Rn, uint32_t Rm, uint32_t Rd, char *dir_flag_V)

flag_V

prototipo de la funcion flag_V que va a operar el registro de la bandera V

void CMN (uint32_t Rn, uint32_t Rm, char *dir_flags)

CMN

prototipo de la funcion CMN que va a operar los registros

void CMP (uint32_t Rn, uint32_t Rm, char *dir_flags)

CMP

prototipo de la funcion CMP que va a operar los registros

```
uint32_t MUL (uint32_t Rn, uint32_t Rm, char *dir_flags)
                                                            MUL
      prototipo de la funcion MUL que va a operar los registros
void TST (uint32_t Rn, uint32_t Rm, char *dir_flags)
                                                             TST
      prototipo de la funcion TST que va a operar los registros
■ uint32 t ADD (uint32 t Rn, uint32 t Rm, char *dir flags)
                                                            ADD
      prototipo de la funcion ADD que va a operar los registros
uint32_t ADC (uint32_t Rn, uint32_t Rm, char *dir_flags)
                                                            ADC
      prototipo de la funcion ADC que va a operar los registros
uint32_t SUB (uint32_t Rn, uint32_t Rm, char *dir_flags)
                                                            SUB
      prototipo de la funcion SUB que va a operar los registros
■ uint32 t SBC (uint32 t Rn, uint32 t Rm, char *dir flags)
                                                            SBC
      prototipo de la funcion SBC que va a operar los registros
```

uint32_t MOV (uint32_t Rn, char *dir_flags)

MOV

prototipo de la funcion MOV que va a operar los registros

uint32_t AND (uint32_t Rn, uint32_t Rm, char *dir_flags)

AND

prototipo de la funcion AND que va a operar los registros

uint32_t EOR (uint32_t Rn, uint32_t Rm, char *dir_flags)

EOR

prototipo de la funcion EOR que va a operar los registros

uint32_t ORR (uint32_t Rn, uint32_t Rm, char *dir_flags)

ORR

prototipo de la funcion ORR que va a operar los registros

void NOP (uint32_t *dir_reg)

NOP

prototipo de la funcion NOP que va a operar los registros

■ void PUSH (uint8 t *SRAM, uint32 t *dir reg, uint8 t *R activos)

Función PUSH

Guarda datos en la Pila

■ uint32 t bitcount (uint8 t *R)

bitcount

funcion bitcount que va a operar los registros contando cuantos estan en 1

void POP (uint8_t *SRAM, uint32_t *dir_reg, uint8_t *R_activos)

POF

Extrae los registros de la pila

uint32_t LDR (uint32_t Rn, uint32_t Rm, uint8_t *SRAM)

Extrae 32bits de SRAM, los guarda en un registro de 32 bits haciendo extension de cero.

uint32_t LDRB (uint32_t Rn, uint32_t Rm, uint8_t *SRAM)

Extrae 8 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de cero.

uint32_t LDRH (uint32_t Rn, uint32_t Rm, uint8_t *SRAM)

Extrae 16 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de cero.

uint32_t LDRSB (uint32_t Rn, uint32_t Rm, uint8_t *SRAM)

Extrae 8 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de signo.

■ uint32 t LDRSH (uint32 t Rn, uint32 t Rm, uint8 t *SRAM)

Extrae 16 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de signo.

void STR (uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t *SRAM)

Guarda 32 bits de un registro en SRAM.

void STRB (uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t *SRAM)

Guarda los primeros 8 bits de un registro en SRAM.

void STRH (uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t *SRAM)

Guarda los primeros 16 bits de un registro en SRAM.

void PUSHI (uint8_t *SRAM, uint32_t *dir_reg, char *dir_flags)

Guarda datos especificos en la pila que son dados por las interrupciones.

void POPI (uint8_t *SRAM, uint32_t *dir_reg, char *dir_flags)

Extrae datos especificos despues de la interrupcion.

void NVIC (uint8_t *IRQ, uint8_t *SRAM, uint32_t *dir_reg, char *dir_flags)
Interrupciones.

5.7.1. Documentación de los 'defines'

5.7.1.1. #define C 2

Macro que define la posicion de la bandera *C* en el puntero dir_flags.

5.7.1.2. #define LR 14

Macro que define la posicion del registro *LR* en el puntero dir_reg.

5.7.1.3. #define N 0

Macro que define la posicion de la bandera *N en* el puntero **dir_flags**.

5.7.1.4. #define PC 15

Macro que define la posicion del registro *PC* en el puntero dir_reg.

5.7.1.5. #define SP 13

Macro que define la posicion del registro SP en el puntero dir_reg.

5.7.1.6. #define V 3

Macro que define la posicion de la bandera *V en* el puntero dir_flags.

5.7.1.7. #define Z 1

Macro que define la posicion de la bandera *Z en* el puntero dir_flags.

5.7.2. Documentación de las funciones

5.7.2.1. uint32_t ADC (uint32_t Rn, uint32_t Rm, char * dir_flags)

ADC

prototipo de la funcion ADC que va a operar los registros

Parámetros

Rn	operando ingresado
_	
Rm	operando ingresaaado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la ADC

5.7.2.2. uint32_t ADD (uint32_t Rn, uint32_t Rm, char * dir_flags)

ADD

prototipo de la funcion ADD que va a operar los registros

Rn	operando ingresado
Rm	operando ingresaaado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la ADD

5.7.2.3. uint32_t AND (uint32_t Rn, uint32_t Rm, char * dir_flags)

AND

prototipo de la funcion AND que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion AND

5.7.2.4. uint32_t bitcount (uint8_t * R)

bitcount

funcion bitcount que va a operar los registros contando cuantos estan en 1

Parámetros

R	registros que va a operar la funcion
---	--------------------------------------

Devuelve

retorna el numero de registro activos

5.7.2.5. void CMN (uint32_t Rn, uint32_t Rm, char * dir_flags)

CMN

prototipo de la funcion CMN que va a operar los registros

	Rn	operando ingresado
	Rm	operando ingresado
dir_	flags	puntero que tiene la direccion de las banderas

5.7.2.6. void CMP (uint32_t Rn, uint32_t Rm, char * dir_flags)

CMP

prototipo de la funcion CMP que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas
un_nags	puntero que tiene la dirección de las banderas

5.7.2.7. uint32_t EOR (uint32_t Rn, uint32_t Rm, char * dir_flags)

EOR

prototipo de la funcion EOR que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion EOR

5.7.2.8. void flag_C (uint32_t Rn, uint32_t Rm, uint32_t Rd, char * dir_flag_C)

flag_C

prototipo de la funcion flag_C que va a operar el registro de la bandera C

Rn	operando ingresado
Rm	operando ingresado
Rd	operando ingresado
dir_flag_C	puntero que tiene la direccion de las banderas

5.7.2.9. void flag_N (uint32_t Rd, char * dir_flag_N)

flag_N

prototipo de la funcion flag_N que va a operar el registro de la bandera N

Parámetros

Rd	operando ingresado
dir_flag_N	puntero que tiene la direccion de las banderas

5.7.2.10. void flag_V (uint32_t Rm, uint32_t Rm, uint32_t Rd, char * dir_flag_V)

flag_V

prototipo de la funcion flag_V que va a operar el registro de la bandera V

Parámetros

Rn	operando ingresado
Rm	operando ingresado
Rd	operando ingresado
dir_flag_V	puntero que tiene la direccion de las banderas

5.7.2.11. void flag_Z (uint32_t Rd, char * dir_flag_Z)

flag_Z

prototipo de la funcion flag_Z que va a operar el registro de la bandera Z

Rd	operando ingresado
dir_flag_Z	puntero que tiene la direccion de las banderas

5.7.2.12. void flags (uint32_t Rn, uint32_t Rm, uint32_t Rd, char * dir_flags)

flags

prototipo de la funcion flags que va a operar los registros de las banderas

Parámetros

Rn	operando ingresado
Rm	operando ingresado
	operation ingroduct
Rd	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

5.7.2.13. uint32_t LDR (uint32_t Rn, uint32_t Rm, uint8_t * SRAM)

Extrae 32bits de SRAM, los guarda en un registro de 32 bits haciendo extension de cero.

Parámetros

Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

Devuelve

Rt registro donde guarda los datos extraidos

5.7.2.14. uint32_t LDRB (uint32_t Rn, uint32_t Rm, uint8_t * SRAM)

Extrae 8 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de cero.

Parámetros

Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

Devuelve

Rt registro donde guarda los datos extraidos

5.7.2.15. uint32_t LDRH (uint32_t Rn, uint32_t Rm, uint8_t * SRAM)

Extrae 16 bits de SRAM,los guarda en un registro de 32 bits haciendo extension de cero.

Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

Devuelve

Rt registro donde guarda los datos extraidos

5.7.2.16. uint32_t LDRSB (uint32_t Rm, uint32_t Rm, uint8_t * SRAM)

Extrae 8 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de signo.

Parámetros

Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

Devuelve

Rt registro donde guarda los datos extraidos

5.7.2.17. uint32_t LDRSH (uint32_t Rn, uint32_t Rm, uint8_t * SRAM)

Extrae 16 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de signo.

Parámetros

Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

Devuelve

Rt registro donde guarda los datos extraidos

5.7.2.18. uint32_t MOV (uint32_t Rn, char * dir_flags)

MOV

prototipo de la funcion MOV que va a operar los registros

Parámetros

Rn	operando ingresado

dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion MOV

5.7.2.19. uint32_t MUL (uint32_t Rn, uint32_t Rm, char * dir_flags)

MUL

prototipo de la funcion MUL que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresado
dia flace	
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion de las MUL

5.7.2.20. void NOP (uint32_t * dir_reg)

NOP

prototipo de la funcion NOP que va a operar los registros

Parámetros

dir_reg	puntero que tiene la direccion de los registros

5.7.2.21. void NVIC (uint8_t * IRQ, uint8_t * SRAM, uint32_t * dir_reg, char * dir_flags)

Interrupciones.

Parámetros

IRQ	puntero al arreglo de interrupciones

SRAM	puntero a la memoria SRAM
dir_reg	puntero a los registro principales
dir_flags	puntero a las banderas

5.7.2.22. uint32_t ORR (uint32_t Rn, uint32_t Rm, char * dir_flags)

ORR

prototipo de la funcion ORR que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion ORR

5.7.2.23. void POP (uint8_t * SRAM, uint32_t * dir_reg, uint8_t * R_activos)

POP

Extrae los registros de la pila

Parámetros

SRAM	registros de SRAM que va a operar la funcion
dir_reg	puntero a registros
R_activos	puntero a registros activos

5.7.2.24. void POPI (uint8_t * SRAM, uint32_t * dir_reg, char * dir_flags)

Extrae datos especificos despues de la interrupcion.

Parámetros

SRAM	puntero a la memoria SRAM
dir_reg	puntero a los registro principales
dir_flags	puntero a las banderas

5.7.2.25. void PUSH (uint8_t * SRAM, uint32_t * dir_reg, uint8_t * R_activos)

Función PUSH

Guarda datos en la Pila

SRAM	puntero a los registros de esta memoria
dir_reg	puntero a los registro principales
R_activos	puntero a los registros activos

5.7.2.26. void PUSHI (uint8 $_{ ext{t}}*$ SRAM, uint32 $_{ ext{t}}*$ dir $_{ ext{reg}}$, char * dir $_{ ext{flags}}$)

Guarda datos especificos en la pila que son dados por las interrupciones.

Parámetros

SRAM	puntero a la memoria SRAM
dir_reg	puntero a los registro principales
dir_flags	puntero a las banderas

5.7.2.27. uint32_t SBC (uint32_t Rn, uint32_t Rm, char * dir_flags)

SBC

prototipo de la funcion SBC que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresaaado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la SBC

5.7.2.28. void STR (uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t * SRAM)

Guarda 32 bits de un registro en SRAM.

Parámetros

Rt	registro a guardar en SRAM
Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

5.7.2.29. void STRB (uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t * SRAM)

Guarda los primeros 8 bits de un registro en SRAM.

Rt	registro a guardar en SRAM
Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

5.7.2.30. void STRH (uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t * SRAM)

Guarda los primeros 16 bits de un registro en SRAM.

Parámetros

Rt	registro a guardar en SRAM
Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

5.7.2.31. uint32_t SUB (uint32_t Rn, uint32_t Rm, char * dir_flags)

SUB

prototipo de la funcion SUB que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la SUB

5.7.2.32. void TST (uint32_t Rn, uint32_t Rm, char * dir_flags)

TST

prototipo de la funcion TST que va a operar los registros

Rn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

5.8. Referencia del Archivo instrucciones.h

```
#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
```

Funciones

void flags (uint32_t Rn, uint32_t Rm, uint32_t Rd, char *dir_flags)flags

- 3 -

prototipo de la funcion flags que va a operar los registros de las banderas

void flag_N (uint32_t Rd, char *dir_flag_N)

flag_N

prototipo de la funcion flag_N que va a operar el registro de la bandera N

void flag_Z (uint32_t Rd, char *dir_flag_Z)

flag_Z

prototipo de la funcion flag_Z que va a operar el registro de la bandera Z

void flag_C (uint32_t Rn, uint32_t Rm, uint32_t Rd, char *dir_flag_C)

flag_C

prototipo de la funcion flag_C que va a operar el registro de la bandera C

void flag_V (uint32_t Rn, uint32_t Rm, uint32_t Rd, char *dir_flag_V)

flag_V

prototipo de la funcion flag_V que va a operar el registro de la bandera V

void CMN (uint32_t Rn, uint32_t Rm, char *dir_flags)

CMN

prototipo de la funcion CMN que va a operar los registros

void CMP (uint32_t Rn, uint32_t Rm, char *dir_flags)

CMP

prototipo de la funcion CMP que va a operar los registros

■ uint32_t MUL (uint32_t Rn, uint32_t Rm, char *dir_flags)

MUL

prototipo de la funcion MUL que va a operar los registros

void TST (uint32_t Rn, uint32_t Rm, char *dir_flags)

TST

prototipo de la funcion TST que va a operar los registros

uint32_t ADD (uint32_t Rn, uint32_t Rm, char *dir_flags)

ADD

prototipo de la funcion ADD que va a operar los registros

uint32_t ADC (uint32_t Rn, uint32_t Rm, char *dir_flags)

ADC

prototipo de la funcion ADC que va a operar los registros

uint32_t SUB (uint32_t Rn, uint32_t Rm, char *dir_flags)

SUB

prototipo de la funcion SUB que va a operar los registros

uint32_t SBC (uint32_t Rn, uint32_t Rm, char *dir_flags)

SBC

prototipo de la funcion SBC que va a operar los registros

uint32_t MOV (uint32_t Rn, char *dir_flags)

MOV

prototipo de la funcion MOV que va a operar los registros

■ uint32_t AND (uint32_t Rn, uint32_t Rm, char *dir_flags)

AND

prototipo de la funcion AND que va a operar los registros

uint32_t EOR (uint32_t Rn, uint32_t Rm, char *dir_flags)

EOR

prototipo de la funcion EOR que va a operar los registros

uint32_t ORR (uint32_t Rn, uint32_t Rm, char *dir_flags)

ORE

prototipo de la funcion ORR que va a operar los registros

void NOP (uint32 t *dir reg)

NOP

prototipo de la funcion NOP que va a operar los registros

void PUSH (uint8_t *SRAM, uint32_t *dir_reg, uint8_t *R_activos)

Función PUSH

Guarda datos en la Pila

uint32_t bitcount (uint8_t *R)

bitcount

funcion bitcount que va a operar los registros contando cuantos estan en 1

void POP (uint8_t *SRAM, uint32_t *dir_reg, uint8_t *R_activos)

POP

Extrae los registros de la pila

uint32_t LDR (uint32_t Rn, uint32_t Rm, uint8_t *SRAM)

Extrae 32bits de SRAM, los guarda en un registro de 32 bits haciendo extension de cero.

uint32 t LDRB (uint32 t Rn, uint32 t Rm, uint8 t *SRAM)

Extrae 8 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de cero.

uint32 t LDRH (uint32 t Rn, uint32 t Rm, uint8 t *SRAM)

Extrae 16 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de cero.

uint32_t LDRSB (uint32_t Rn, uint32_t Rm, uint8_t *SRAM)

Extrae 8 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de signo.

uint32_t LDRSH (uint32_t Rn, uint32_t Rm, uint8_t *SRAM)

Extrae 16 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de signo.

void STR (uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t *SRAM)

Guarda 32 bits de un registro en SRAM.

■ void STRB (uint32 t Rt, uint32 t Rn, uint32 t Rm, uint8 t *SRAM)

Guarda los primeros 8 bits de un registro en SRAM.

void STRH (uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t *SRAM)

Guarda los primeros 16 bits de un registro en SRAM.

void NVIC (uint8_t *IRQ, uint8_t *SRAM, uint32_t *dir_reg, char *dir_flags)

Interrupciones.

void PUSHI (uint8_t *SRAM, uint32_t *dir_reg, char *dir_flags)

Guarda datos especificos en la pila que son dados por las interrupciones.

void POPI (uint8_t *SRAM, uint32_t *dir_reg, char *dir_flags)

Extrae datos especificos despues de la interrupcion.

5.8.1. Documentación de las funciones

5.8.1.1. uint32_t ADC (uint32_t Rn, uint32_t Rm, char * dir_flags)

ADC

prototipo de la funcion ADC que va a operar los registros

	Rn	operando ingresado
	Rm	operando ingresaaado
dir_fi	lags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la ADC

5.8.1.2. uint32_t ADD (uint32_t Rn, uint32_t Rm, char * dir_flags)

ADD

prototipo de la funcion ADD que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresaaado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la ADD

5.8.1.3. uint32_t AND (uint32_t Rn, uint32_t Rm, char * dir_flags)

AND

prototipo de la funcion AND que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion AND

5.8.1.4. uint32_t bitcount (uint8_t * R)

bitcount

funcion bitcount que va a operar los registros contando cuantos estan en 1

Parámetros

D	registros que va a operar la funcion
П	registros que va a operar la funcion

Devuelve

retorna el numero de registro activos

5.8.1.5. void CMN (uint32_t Rn, uint32_t Rm, char * dir_flags)

CMN

prototipo de la funcion CMN que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

5.8.1.6. void CMP (uint32_t Rn, uint32_t Rm, char * dir_flags)

CMP

prototipo de la funcion CMP que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

5.8.1.7. uint32_t EOR (uint32_t Rn, uint32_t Rm, char * dir_flags)

EOR

prototipo de la funcion EOR que va a operar los registros

Rn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion EOR

5.8.1.8. void flag_C (uint32_t Rn, uint32_t Rm, uint32_t Rd, char * dir_flag_C)

flag_C

prototipo de la funcion flag_C que va a operar el registro de la bandera C

Parámetros

Rn	operando ingresado
Rm	operando ingresado
Rd	operando ingresado
dir_flag_C	puntero que tiene la direccion de las banderas

5.8.1.9. void flag_N (uint32_t Rd, char * dir_flag_N)

flag_N

prototipo de la funcion flag_N que va a operar el registro de la bandera N

Parámetros

Rd	operando ingresado
dir_flag_N	puntero que tiene la direccion de las banderas

5.8.1.10. void flag_V (uint32_t Rm, uint32_t Rm, uint32_t Rd, char * dir_flag_V)

flag_V

prototipo de la funcion flag_V que va a operar el registro de la bandera V

Rn	operando ingresado
Rm	operando ingresado
Rd	operando ingresado
dir_flag_V	puntero que tiene la direccion de las banderas

5.8.1.11. void flag_Z (uint32_t Rd, char * dir_flag_Z)

flag Z

prototipo de la funcion flag_Z que va a operar el registro de la bandera Z

Parámetros

Rd	operando ingresado
dir_flag_Z	puntero que tiene la direccion de las banderas

5.8.1.12. void flags (uint32_t Rn, uint32_t Rm, uint32_t Rd, char * dir_flags)

flags

prototipo de la funcion flags que va a operar los registros de las banderas

Parámetros

Rn	operando ingresado
Rm	operando ingresado
Rd	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

5.8.1.13. uint32_t LDR (uint32_t Rn, uint32_t Rm, uint8_t * SRAM)

Extrae 32bits de SRAM, los guarda en un registro de 32 bits haciendo extension de cero.

Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

Devuelve

Rt registro donde guarda los datos extraidos

5.8.1.14. uint32_t LDRB (uint32_t Rn, uint32_t Rm, uint8_t * SRAM)

Extrae 8 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de cero.

Parámetros

Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

Devuelve

Rt registro donde guarda los datos extraidos

5.8.1.15. uint32_t LDRH (uint32_t Rn, uint32_t Rm, uint8_t * SRAM)

Extrae 16 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de cero.

Parámetros

Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

Devuelve

Rt registro donde guarda los datos extraidos

5.8.1.16. uint32_t LDRSB (uint32_t Rm, uint32_t Rm, uint8_t * SRAM)

Extrae 8 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de signo.

Parámetros

Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

Devuelve

Rt registro donde guarda los datos extraidos

5.8.1.17. uint32_t LDRSH (uint32_t Rn, uint32_t Rm, uint8_t * SRAM)

Extrae 16 bits de SRAM, los guarda en un registro de 32 bits haciendo extension de signo.

Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

Devuelve

Rt registro donde guarda los datos extraidos

5.8.1.18. uint32_t MOV (uint32_t Rn, char * dir_flags)

MOV

prototipo de la funcion MOV que va a operar los registros

Parámetros

Rn	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion MOV

5.8.1.19. uint32_t MUL (uint32_t Rn, uint32_t Rm, char * dir_flags)

MUL

prototipo de la funcion MUL que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion de las MUL

5.8.1.20. void NOP (uint32_t * dir_reg)

NOP

prototipo de la funcion NOP que va a operar los registros

Parámetros

dir_reg	puntero que tiene la direccion de los registros

5.8.1.21. void NVIC (uint8_t * IRQ, uint8_t * SRAM, uint32_t * dir_reg , char * dir_flags)

Interrupciones.

Parámetros

IRQ	puntero al arreglo de interrupciones
SRAM	puntero a la memoria SRAM
dir_reg	puntero a los registro principales
dir_flags	puntero a las banderas

5.8.1.22. uint32_t ORR (uint32_t Rn, uint32_t Rm, char * dir_flags)

ORR

prototipo de la funcion ORR que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la operacion ORR

5.8.1.23. void POP (uint8_t * SRAM, uint32_t * dir_reg, uint8_t * R_activos)

POP

Extrae los registros de la pila

Parámetros

Generado el Domingo, 18 de Octubre de 2015 15:31:05 para Emulador Cortex-M0 por Doxygen

SRAM	registros de SRAM que va a operar la funcion
dir_reg	puntero a registros
R_activos	puntero a registros activos

5.8.1.24. void POPI (uint8_t * SRAM, uint32_t * dir_reg, char * dir_flags)

Extrae datos especificos despues de la interrupcion.

Parámetros

SRAM	puntero a la memoria SRAM
dir_reg	puntero a los registro principales
dir_flags	puntero a las banderas

5.8.1.25. void PUSH (uint8_t * SRAM, uint32_t * dir_reg, uint8_t * R_activos)

Función PUSH

Guarda datos en la Pila

Parámetros

SRAM	puntero a los registros de esta memoria
dir_reg	puntero a los registro principales
R_activos	puntero a los registros activos

5.8.1.26. void PUSHI (uint8_t * SRAM, uint32_t * dir_reg, char * dir_flags)

Guarda datos especificos en la pila que son dados por las interrupciones.

Parámetros

SRAM	puntero a la memoria SRAM
dir_reg	puntero a los registro principales
dir_flags	puntero a las banderas

5.8.1.27. uint32_t SBC (uint32_t Rn, uint32_t Rm, char * dir_flags)

SBC

prototipo de la funcion SBC que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresaaado

dir_flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la SBC

5.8.1.28. void STR (uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t * SRAM)

Guarda 32 bits de un registro en SRAM.

Parámetros

Rt	registro a guardar en SRAM
Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

5.8.1.29. void STRB (uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t * SRAM)

Guarda los primeros 8 bits de un registro en SRAM.

Parámetros

Rt	registro a guardar en SRAM
Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

5.8.1.30. void STRH (uint32_t Rt, uint32_t Rn, uint32_t Rm, uint8_t * SRAM)

Guarda los primeros 16 bits de un registro en SRAM.

Parámetros

Rt	registro a guardar en SRAM
Rn	direccion de memoria
Rm	numero o direccion de memoria
SRAM	puntero a la memoria SRAM

5.8.1.31. uint32_t SUB (uint32_t Rn, uint32_t Rm, char * dir_flags)

SUB

prototipo de la funcion SUB que va a operar los registros

	Rn	operando ingresado
	Rm	operando ingresado
dir_	flags	puntero que tiene la direccion de las banderas

Devuelve

retorna el resultado de la SUB

5.8.1.32. void TST (uint32_t Rn, uint32_t Rm, char * dir_flags)

TST

prototipo de la funcion TST que va a operar los registros

Parámetros

Rn	operando ingresado
Rm	operando ingresado
dir_flags	puntero que tiene la direccion de las banderas

5.9. Referencia del Archivo io.c

#include "io.h"

Funciones

void initIO (void)

Inicializa los puertos de Entrada y Salida.

void changePinPortA (uint8_t pin, uint8_t value)

Cambia el valor de los puertos A.

void changePinPortB (uint8_t pin, uint8_t value)

Cambia el valor de los puertos B.

void IOAccess (uint8_t address, uint8_t *data, uint8_t r_w)

Escribe datos en el DDRB.

void showPorts (void)

Muestra los puertos en la interfaz.

■ void showFrame (int x, int y, int w, int h)

Organiza la interfaz de los puertos.

Variables

- port_t PORTA
- port_t PORTB

5.9.1. Documentación de las funciones

5.9.1.1. void changePinPortA (uint8_t pin, uint8_t value)

Cambia el valor de los puertos A.

Parámetros

pin	numero del pin
value	valor del pin

5.9.1.2. void changePinPortB (uint8_t pin, uint8_t value)

Cambia el valor de los puertos B.

Parámetros

ſ	pin	numero del pin
	value	valor del pin

5.9.1.3. void initlO (void)

Inicializa los puertos de Entrada y Salida.

5.9.1.4. void IOAccess (uint8_t address, uint8_t * data, uint8_t r_w)

Escribe datos en el DDRB.

Parámetros

address	direccion de memoria
data	puntero al dato
r_w	lee o escribe

5.9.1.5. void showFrame (int x, int y, int w, int h)

Organiza la interfaz de los puertos.

Parámetros

X	coordenada X
у	coordenada Y
W	
h	

5.9.1.6. void showPorts (void)

Muestra los puertos en la interfaz.

5.9.2. Documentación de las variables

```
5.9.2.1. port_t PORTA
```

```
5.9.2.2. port_t PORTB
```

5.10. Referencia del Archivo io.h

```
#include <stdint.h>
#include <curses.h>
```

Estructuras de datos

struct port_t

'defines'

- #define XINIT 10
- #define YINIT 5
- #define HIGH 1
- #define LOW 0
- #define Read 1
- #define Write 0
- #define BLUEBLACK 10 /*Text Blue Background Black*/
- #define REDBLACK 20 /*Text Red Background Black*/
- #define WHITEBLACK 30 /*Text White Background White*/

Funciones

■ void IOAccess (uint8_t address, uint8_t *data, uint8_t r_w)

void changePinPortA (uint8_t pin, uint8_t value)

Cambia el valor de los puertos A.

Escribe datos en el DDRB.

void changePinPortB (uint8_t pin, uint8_t value)

Cambia el valor de los puertos B.

void initIO (void)

Inicializa los puertos de Entrada y Salida.

void showPorts (void)

Muestra los puertos en la interfaz.

void showFrame (int x, int y, int w, int h)

Organiza la interfaz de los puertos.

Variables

uint8_t IRQ [16]

5.10.1. Documentación de los 'defines'

5.10.1.1. #define BLUEBLACK 10 /*Text Blue Background Black*/

5.10.1.2. #define HIGH 1

5.10.1.3. #define LOW 0

5.10.1.4. #define Read 1

5.10.1.5. #define REDBLACK 20 /*Text Red Background Black*/

5.10.1.6. #define WHITEBLACK 30 /*Text White Background White*/

5.10.1.7. #define Write 0

5.10.1.8. #define XINIT 10

5.10.1.9. #define YINIT 5

5.10.2. Documentación de las funciones

5.10.2.1. void changePinPortA (uint8_t pin, uint8_t value)

Cambia el valor de los puertos A.

Parámetros

pin	numero del pin
value	valor del pin

5.10.2.2. void changePinPortB (uint8_t pin, uint8_t value)

Cambia el valor de los puertos B.

Parámetros

pin	numero del pin
value	valor del pin

5.10.2.3. void initlO (void)

Inicializa los puertos de Entrada y Salida.

5.10.2.4. void IOAccess (uint8_t address, uint8_t * data, uint8_t r_w)

Escribe datos en el DDRB.

Parámetros

address	direccion de memoria
data	puntero al dato

	la a a a a will a
r w	lee o escripe
'-"	100 0 0001100

5.10.2.5. void showFrame (int x, int y, int w, int h)

Organiza la interfaz de los puertos.

Parámetros

X	coordenada X
У	coordenada Y
W	
h	

5.10.2.6. void showPorts (void)

Muestra los puertos en la interfaz.

5.10.3. Documentación de las variables

5.10.3.1. uint8_t IRQ[16]

5.11. Referencia del Archivo main.c

```
#include <curses.h>
#include "imprimir.h"
#include "instrucciones.h"
#include "instruc_desplazamiento.h"
#include "salto.h"
#include "decoder.h"
#include "io.h"
#include <time.h>
```

'defines'

■ #define N 0

Macro que define la posicion de la bandera N en el registroAPSR.

#define Z 1

Macro que define la posicion de la bandera Z en el registroAPSR.

#define C 2

Macro que define la posicion de la bandera C en el registro APSR.

■ #define V 3

Macro que define la posicion de la bandera V en el registro APSR.

■ #define LR 14

Macro que define la posicion del registro LR en el arreglo R.

■ #define PC 15

Macro que define la posicion del registro PC en el arreglo R.

#define SP 13

Macro que define la posicion del registro SP en el arreglo R.

#define TAM_SRAM 255

Macro que define el tamaño del arreglo que emula la memoria **SRAM**.

Funciones

■ int main (void)

5.11.1. Documentación de los 'defines'

5.11.1.1. #define C 2

Macro que define la posicion de la bandera *C en* el registro *APSR*.

5.11.1.2. #define LR 14

Macro que define la posicion del registro *LR* en el arreglo *R*.

5.11.1.3. #define N 0

Macro que define la posicion de la bandera N en el registro APSR.

5.11.1.4. #define PC 15

Macro que define la posicion del registro *PC en* el arreglo *R*.

5.11.1.5. #define SP 13

Macro que define la posicion del registro SP en el arreglo R.

5.11.1.6. #define TAM_SRAM 255

Macro que define el tamaño del arreglo que emula la memoria SRAM.

5.11.1.7. #define V 3

Macro que define la posicion de la bandera *V en* el registro *APSR*.

5.11.1.8. #define Z 1

Macro que define la posicion de la bandera *Z en* el registro*APSR*.

5.11.2. Documentación de las funciones

5.11.2.1. int main (void)

5.12. Referencia del Archivo README.md

5.13. Referencia del Archivo salto.c

#include "salto.h"

'defines'

#define N 0

Macro que define la posicion de la bandera N en el puntero dir_flags.

#define Z 1

Macro que define la posicion de la bandera Z en el puntero dir_flags.

#define C 2

Macro que define la posicion de la bandera C en el puntero dir_flags.

#define V 3

Macro que define la posicion de la bandera V en el puntero dir_flags.

■ #define LR 14

Macro que define la posicion del registro LR en el puntero dir_reg.

■ #define PC 15

Macro que define la posicion del registro PC en el puntero dir_reg.

Funciones

void B (uint32_t label, uint32_t *dir_reg)

В

prototipo de la funcion B que va a modificar el registro PC

void BL (uint32_t label, uint32_t *dir_reg)

BL

prototipo de la funcion BL que va a modificar el registro PC

void BX (uint32_t *dir_reg)

ВХ

prototipo de la funcion BX que va a modificar el registro PC

■ void BEQ (uint32 t label, uint32 t *dir reg, char *dir flags)

BEQ

prototipo de la funcion BEQ que va a modificar el registro PC

■ void BNE (uint32 t label, uint32 t *dir reg, char *dir flags)

BNE

prototipo de la funcion BNE que va a modificar el registro PC

void BCS (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BCS

prototipo de la funcion BCS que va a modificar el registro PC

void BCC (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BCC

prototipo de la funcion BCC que va a modificar el registro PC

void BMI (uint32_t label, uint32_t *dir_reg, char *dir_flags)
RMI

prototipo de la funcion BMI que va a modificar el registro PC

void BPL (uint32_t label, uint32_t *dir_reg, char *dir_flags)
BPL

prototipo de la funcion BPL que va a modificar el registro PC

void BVS (uint32_t label, uint32_t *dir_reg, char *dir_flags)BVS

prototipo de la funcion BVS que va a modificar el registro PC

void BVC (uint32_t label, uint32_t *dir_reg, char *dir_flags)
BVC

prototipo de la funcion BVC que va a modificar el registro PC

void BHI (uint32_t label, uint32_t *dir_reg, char *dir_flags)
BHI

prototipo de la funcion BHI que va a modificar el registro PC

void BLS (uint32_t label, uint32_t *dir_reg, char *dir_flags)
BLS

prototipo de la funcion BLS que va a modificar el registro PC

void BGE (uint32_t label, uint32_t *dir_reg, char *dir_flags)
BGE

prototipo de la funcion BGE que va a modificar el registro PC

void BLT (uint32_t label, uint32_t *dir_reg, char *dir_flags)BLT

prototipo de la funcion BLT que va a modificar el registro PC

void BGT (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BGT

prototipo de la funcion BGT que va a modificar el registro PC

void BLE (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BLE

prototipo de la funcion BLE que va a modificar el registro PC

void BAL (uint32 t label, uint32 t *dir reg)

BAL

prototipo de la funcion BAL que va a modificar el registro PC

5.13.1. Documentación de los 'defines'

5.13.1.1. #define C 2

Macro que define la posicion de la bandera *C en* el puntero dir_flags.

5.13.1.2. #define LR 14

Macro que define la posicion del registro *LR* en el puntero dir_reg.

5.13.1.3. #define N 0

Macro que define la posicion de la bandera $\it N$ en el puntero $\it dir_flags$.

5.13.1.4. #define PC 15

Macro que define la posicion del registro *PC en* el puntero dir_reg.

5.13.1.5. #define V 3

Macro que define la posicion de la bandera *V en* el puntero dir_flags.

5.13.1.6. #define Z 1

Macro que define la posicion de la bandera Z en el puntero dir_flags.

5.13.2. Documentación de las funciones

5.13.2.1. void B (uint32_t *label*, uint32_t * *dir_reg*)

В

prototipo de la funcion B que va a modificar el registro PC

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros

5.13.2.2. void BAL (uint32_t *label*, uint32_t * *dir_reg*)

BAL

prototipo de la funcion BAL que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros

5.13.2.3. void BCC (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BCC

prototipo de la funcion BCC que va a modificar el registro PC

Parámetros

label	operando ingresado
dir roa	and the state of t
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.13.2.4. void BCS (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BCS

prototipo de la funcion BCS que va a modificar el registro PC

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.13.2.5. void BEQ (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BEQ

prototipo de la funcion BEQ que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.13.2.6. void BGE (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BGE

prototipo de la funcion BGE que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.13.2.7. void BGT (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BGT

prototipo de la funcion BGT que va a modificar el registro PC

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.13.2.8. void BHI (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BHI

prototipo de la funcion BHI que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.13.2.9. void BL (uint32_t label, uint32_t * dir_reg)

BL

prototipo de la funcion BL que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros

5.13.2.10. void BLE (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BLE

prototipo de la funcion BLE que va a modificar el registro PC

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.13.2.11. void BLS (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BLS

prototipo de la funcion BLS que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.13.2.12. void BLT (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BLT

prototipo de la funcion BLT que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.13.2.13. void BMI (uint32_t label, uint32_t * dir_reg, char * dir_flags)

ВМІ

prototipo de la funcion BMI que va a modificar el registro PC

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.13.2.14. void BNE (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BNE

prototipo de la funcion BNE que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.13.2.15. void BPL (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BPL

prototipo de la funcion BPL que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.13.2.16. void BVC (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BVC

prototipo de la funcion BVC que va a modificar el registro PC

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.13.2.17. void BVS (uint32_t label, uint32_t * dir_reg , char * dir_flags)

BVS

prototipo de la funcion BVS que va a modificar el registro PC

Parámetros

labe	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.13.2.18. void BX (uint32_t * dir_reg)

 BX

prototipo de la funcion BX que va a modificar el registro PC

Parámetros

dir_reg	puntero que tiene la direccion de los registros

5.14. Referencia del Archivo salto.h

```
#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
```

Funciones

void B (uint32_t label, uint32_t *dir_reg)

prototipo de la funcion B que va a modificar el registro PC

void BL (uint32_t label, uint32_t *dir_reg)

BL

prototipo de la funcion BL que va a modificar el registro PC

void BX (uint32_t *dir_reg)

вх

prototipo de la funcion BX que va a modificar el registro PC

void BEQ (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BEQ

prototipo de la funcion BEQ que va a modificar el registro PC

■ void BNE (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BNE

prototipo de la funcion BNE que va a modificar el registro PC

void BCS (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BCS

prototipo de la funcion BCS que va a modificar el registro PC

void BCC (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BCC

prototipo de la funcion BCC que va a modificar el registro PC

void BMI (uint32_t label, uint32_t *dir_reg, char *dir_flags)

ВМІ

prototipo de la funcion BMI que va a modificar el registro PC

void BPL (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BPL

prototipo de la funcion BPL que va a modificar el registro PC

void BVS (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BVS

prototipo de la funcion BVS que va a modificar el registro PC

void BVC (uint32_t label, uint32_t *dir_reg, char *dir_flags)
BVC

prototipo de la funcion BVC que va a modificar el registro PC

void BHI (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BH

prototipo de la funcion BHI que va a modificar el registro PC

void BLS (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BLS

prototipo de la funcion BLS que va a modificar el registro PC

void BGE (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BGE

prototipo de la funcion BGE que va a modificar el registro PC

void BLT (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BLT

prototipo de la funcion BLT que va a modificar el registro PC

void BGT (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BGT

prototipo de la funcion BGT que va a modificar el registro PC

void BLE (uint32_t label, uint32_t *dir_reg, char *dir_flags)

BLE

prototipo de la funcion BLE que va a modificar el registro PC

void BAL (uint32_t label, uint32_t *dir_reg)

BAL

prototipo de la funcion BAL que va a modificar el registro PC

5.14.1. Documentación de las funciones

5.14.1.1. void B (uint32_t *label*, uint32_t * *dir_reg*)

prototipo de la funcion B que va a modificar el registro PC

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros

5.14.1.2. void BAL (uint32_t label, uint32_t * dir_reg)

BAL

prototipo de la funcion BAL que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros

5.14.1.3. void BCC (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BCC

prototipo de la funcion BCC que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.14.1.4. void BCS (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BCS

prototipo de la funcion BCS que va a modificar el registro PC

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.14.1.5. void BEQ (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BEQ

prototipo de la funcion BEQ que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.14.1.6. void BGE (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BGE

prototipo de la funcion BGE que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.14.1.7. void BGT (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BGT

prototipo de la funcion BGT que va a modificar el registro PC

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.14.1.8. void BHI (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BHI

prototipo de la funcion BHI que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.14.1.9. void BL (uint32_t *label*, uint32_t * *dir_reg*)

BL

prototipo de la funcion BL que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros

5.14.1.10. void BLE (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BLE

prototipo de la funcion BLE que va a modificar el registro PC

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.14.1.11. void BLS (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BLS

prototipo de la funcion BLS que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.14.1.12. void BLT (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BLT

prototipo de la funcion BLT que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.14.1.13. void BMI (uint32_t label, uint32_t * dir_reg, char * dir_flags)

ВМІ

prototipo de la funcion BMI que va a modificar el registro PC

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.14.1.14. void BNE (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BNE

prototipo de la funcion BNE que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.14.1.15. void BPL (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BPL

prototipo de la funcion BPL que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.14.1.16. void BVC (uint32_t label, uint32_t * dir_reg, char * dir_flags)

BVC

prototipo de la funcion BVC que va a modificar el registro PC

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.14.1.17. void BVS (uint32_t label, uint32_t * dir_reg , char * dir_flags)

BVS

prototipo de la funcion BVS que va a modificar el registro PC

Parámetros

label	operando ingresado
dir_reg	puntero que tiene la direccion de los registros
dir_flags	puntero que tiene la direccion de las banderas

5.14.1.18. void BX (uint32_t * dir_reg)

 $\mathbf{B}\mathbf{X}$

prototipo de la funcion BX que va a modificar el registro PC

Parámetros

dir_reg	puntero que tiene la direccion de los registros

5.15. Referencia del Archivo test.c

```
#include "imprimir.h"
#include "instrucciones.h"
#include "instruc_desplazamiento.h"
#include <curses.h>
```

'defines'

■ #define N 0

Macro que define la posicion de la bandera N en el registroregf.

■ #define Z 1

Macro que define la posicion de la bandera Z en el registroregf.

■ #define C 2

Macro que define la posicion de la bandera C en el registro regf.

■ #define V 3

Macro que define la posicion de la bandera V en el registro regf.

Funciones

int main ()

TEXT

Este archivo es para probar las librerias.

5.15.1. Documentación de los 'defines'

5.15.1.1. #define C 2

Macro que define la posicion de la bandera *C en* el registro *regf*.

5.15.1.2. #define N 0

Macro que define la posicion de la bandera N en el registro regf.

5.15.1.3. #define V 3

Macro que define la posicion de la bandera *V en* el registro *regf*.

5.15.1.4. #define Z 1

Macro que define la posicion de la bandera *Z en* el registro*regf*.

5.15.2. Documentación de las funciones

5.15.2.1. int main (void)

TEXT

Este archivo es para probar las librerias.